AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (**Currently Amended**) A system for construction of a customizable service oriented software system and framework, the system comprising:

a server infrastructure comprising a kernel having a hard-coded portion for getting or fetching service interface definitions; and

a set of management and design tools stored on physical computer-readable media that, when executed by the server infrastructure, causes the system to perform management and development of service-oriented software modules as services,

wherein the system uses service-oriented software service modules to perform system functions to enable operation of the system itself, wherein service interface definitions for the service-oriented software service modules that perform system functions are first described using the set of management and design tools and then consumed by the same set of management and design tools,

wherein execution of the system functions includes the service-oriented software service modules of the system functions being implemented through the server infrastructure itself, wherein the kernel of the server infrastructure gets or fetches any service interface definitions for the service-oriented software service modules that perform system functions such that the kernel breaks circularity created when the service interface definitions reference the server infrastructure.

2. (Previously Presented) The system of claim 1, further comprising:

a communication module between the set of software management and design tools developed for supporting the system functions of the system and a runtime platform of the system through a set of service-oriented software service modules implemented through the same set of management and design tools provided to end-users of the system, wherein the communication module accomplishes transparent distribution for parts of the system that are consumers of service-oriented software services from parts of the system that are producers of service-oriented software services.

3. (Previously Presented) The system of claim 1, further comprising:

the server infrastructure and set of management and design tools configured to perform rapid convergence of quality during construction and advancement of the system functions through the same set of management and design tools provided to end-users of the system.

4. (Previously Presented) The system of claim 1, further comprising:

the server infrastructure and set of management and design tools configured to reduce an implementation time of the system functions through the same set of management and design tools provided to end-users of the system.

5. (Previously Presented) The system of claim 1, further comprising:

the server infrastructure and set of management and design tools configured to create a high-degree of customizability through exposure of system functions as consumable service-oriented software service modules through the same set of management and design tools provided to end-users of the system.

6. (Previously Presented) The system of claim 2, further comprising:

a service log manager tool for managing, viewing and analyzing the services dispatched through the system that uses a set of service-oriented software service modules to interact with the system through the same set of management and design tools provided to end-users of the system.

7. (Previously Presented) The system of claim 2, further comprising:

a service manager tool for managing current running services that uses a set of service-oriented software service modules to interact with the system through the same set of management and design tools provided to end-users of the system.

8. (Previously Presented) The system of claim 2, further comprising:

a service cache manager tool for managing cached services within the system that uses a set of service-oriented software service modules to interact with the system through the same set of management and design tools provided to end-users of the system.

9. (Previously Presented) The system of claim 2, further comprising:

a system shared-memory manager tool for managing content of the system shared memory that uses a set of service-oriented software service modules to interact with the system through the same set of management and design tools provided to end-users of the system.

10. (Previously Presented) The system of claim 2, further comprising:

a consumer account provisioning manager tool used to provision and deploy service-oriented solutions that uses a set of service-oriented software service modules to interact with the system through the same set of management and design tools provided to end-users of the system.

11. (Previously Presented) The system of claim 2, further comprising:

a security manager tool used for user and role management that uses a set of service-oriented software service modules to interact with the system through the same set of management and design tools provided to end-users of the system.

12. (Previously Presented) The system of claim 2, further comprising:

a system cluster manager used for load-balancing and managing clusters of the system that uses a set of service-oriented software service modules to interact with the cluster of systems through the same set of management and design tools provided to end-users of the system.

13. (Previously Presented) The system of claim 2, further comprising:

any management or design tool that needs to interact with the system to use a set of service-oriented software service modules to interact with the system through the same set of management and design tools provided to end-users of the system.

14. (Previously Presented) The system of claim 1, wherein the system is extended through a set of service-oriented software service modules to implement the system functions required for supporting the functionality of the system through the same set of management and design tools provided to end-users of the system.

15. (Previously Presented) The system of claim 1, wherein all system functions required for management, design and invocation of system functionality by the system are implemented as service-oriented software service modules through the same set of management and design tools provided to end-users of the system.

16. (Previously Presented) The system of claim 1, wherein the implementation of all of the system functions that are service-oriented software service modules can be replaced transparently for customizing the system functionality using the same set of management and design tools provided to end-users of the system.

Claims 17 through 19. (Cancelled)

20. (**Currently Amended**) In a computer network, a service-oriented development system for the composition and implementation of service-oriented software modules, the service-oriented development system itself being built on top of service-oriented software modules, the service-oriented development system comprising:

a) a user interface tool stored on physical computer-readable media that, when executed by one or more processors, causes the system to allow an end-user to develop, assemble, manage and/or execute implementation of service-oriented software modules; and

b) a run-time server comprising a kernel ~~having a core module~~ that provides a framework utilizing interfaces with pluggable implementations for dispatching the service-oriented software modules, the kernel ~~core module~~ comprising a hard-coded portion for getting or fetching a definition of any service-oriented software module,

each of the user interface tool and run-time server requiring system functions to enable the operation of the service-oriented development system, at least some of the system functions of the user interface tool and run-time server are implemented as service-oriented software service modules using the user interface tool to define definitions of the system functions that are service-oriented software service modules,

wherein the kernel ~~core module~~ is configured to implement the at least some system functions by invoking the service-oriented software service modules of the at least some system functions, wherein the kernel gets or fetches any definition for the service-oriented software service modules that perform system functions such that the kernel breaks circularity created when the definitions reference the run-time server.

21. (Previously Presented) The service-oriented development system as recited in claim 20, wherein the user-interface tool comprises a management tool for allowing an end-user to manage service-oriented software modules.

22. (Previously Presented) The service-oriented development system as recited in claim 20, wherein the at least some of the system functions of the user interface tool and the run-time server being themselves built to use service-oriented software service modules comprises at least one of:

service interface metadata management;
log analyzing;
searching;
service monitoring and management;
cache management;
system configuration;
shared memory management;
event broadcasting and notification;
security management;
provisioning; or
cluster management.

23. (Previously Presented) The service-oriented development system as recited in claim 20, wherein the user interface tool and run-time server allow the end-user to customize, replace, or extend one or more of the at least some of the system functions using the end-user functionality of the service-oriented development system.

24. (Previously Presented) The service-oriented development system as recited in claim 20, wherein the user interface tool and the run-time server allow the end-user to rapidly converge a quality of a software system under construction.

25. (**Currently Amended**) The service-oriented development system as recited in claim 20, wherein the user-interface tool consumes those system functions built as service-oriented software service modules, and wherein implementation of those services are dispatched by the kernel ~~core module~~ and implemented through the same framework that is provided to the end-user for developing, assembling, managing and/or executing implementation of service-oriented software modules.

26. (**Currently Amended**) The service-oriented development system as recited in claim 20, wherein the kernel ~~core module~~ is configured to dispatch service-oriented software modules implementation using a multi-threaded process abstraction.

27. (**Currently Amended**) In a network environment comprising a service-oriented development system for the composition, management, and implementation of service-oriented software modules, the service-oriented development system including a user-interface tool for allowing an end-user to develop, assemble, manage, and[[/or]] execute implementation of service-oriented software modules and including a runtime server for implementing service-oriented software modules, a method for transparently distributing service invocations of service-oriented software modules, the method comprising:

using a user-interface tool of a service-oriented development system to define interface definitions for one or more service-oriented software service modules for performing a system function to enable operation of the service-oriented development system;

using an invoker interface to request the one or more service-oriented software service modules for performing the [[a]] system function to enable operation of the service-oriented development system, wherein, consuming a particular service-oriented software service module comprises generating an invocation request, and sending the invocation request to a kernel of a run-time server, the kernel having a hard-coded portion for getting or fetching service interface definitions ~~core module~~;

at the kernel ~~core module~~, receiving the invocation request from the consumer of the service-oriented software service module, the consumer being the service-oriented development system, wherein the kernel of the run-time server gets or fetches any definitions for the service-oriented software service modules that perform system functions such that the kernel breaks circularity created when the definitions reference the run-time server;

using a local invoker to access a runtime environment in a same address space as the consumer of the service-oriented software service module; and

using a remote invoker to access a runtime environment outside the address space of the consumer of the service-oriented software service module, wherein at least one of the local invoker and remote invoker are used to invoke a service-oriented software service module for performing a system function of the service-oriented development system, wherein operation of the internal invoker and remote invoker are encapsulated from the implementer by the invoker interface such that the consumer is not aware whether the invocation request is being sent via the local invoker or the remote invoker, and wherein the kernel ~~core module~~ can switch between an offline internal invoker mode and one or more servers having remote invokers.

28. (Previously Presented) The method as recited in claim 27, wherein using a remote invoker to access a runtime environment outside the address space of the consumer of the service-oriented software service module further comprises:

serializing the invocation request for the service-oriented software service module;

communicating the serialized invocation request using a network protocol;

receiving a serialized response including outputs related to the service-oriented software service module for which the invocation request was serialized; and

deserializing the serialized response to a native object form of the requesting service-oriented software service module.

29. (**Currently Amended**) The method as recited in claim 28, wherein the kernel ~~core module~~ is the consumer of the service-oriented software service module and the service-oriented software service module is a system function of the kernel ~~core module~~ required to implement functionality of the kernel ~~core module~~.

30. (Previously Presented) The method as recited in claim 29, wherein the system function comprises at least one of:

service interface metadata management;

log analyzing;

searching;

service monitoring and management;

cache management;

system configuration;

shared memory management;

event broadcasting and notification;

security management;

provisioning; or

cluster management.

31. (**Currently Amended**) The method as recited in claim 27, wherein the kernel ~~core module~~ of one instance of the system can access metadata of another remote instance of the system by using a corresponding remote invoker instead of its own internal invoker when consuming metadata data access service interfaces.

32. (Previously Presented) The method as recited in claim 27, further comprising:

identifying addresses of other instances of the system; and

using the remote invoker to broadcast a message to other instances of the system.

33. (**Currently Amended**) The method as recited in claim 32, further comprising:

including in the broadcast message a callback invoker address, such that the other instances of the system can return an invocation of a service to the kernel ~~core module~~.